UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/683,933	10/10/2003	Sunay Tripathi	SUN-P8978	2203

81505 7590 01/05/2010

Sun Microsystems, Inc
c/o Marsh Fischmann & Breyfogle LLP
8055 East Tufts Avenue
Suite 450
Denver, CO 80237

EXAMINER	
DENNISON, JERRY B	

ART UNIT	PAPER NUMBER
2443	

MAIL DATE	DELIVERY MODE
01/05/2010	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 10/683,933
Filing Date: October 10, 2003
Appellant(s): TRIPATHI ET AL.

Kent A. Lembke
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed 9/30/2009 appealing from the Office action mailed 10/24/2009.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

6,338,078	Chang et al.	1-2002
7,076,545	DiMambro	7-2006

2002/0112188 Syvanne 8-2002

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Specification

1. The Specification is objected to because the "Cross Reference to Related Applications" section is incomplete. Correction is required. See MPEP § 608.01(b).

Claim Rejections - 35 USC § 102

The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.
>
> (e) the invention was described in a patent granted on an application for patent by another filed in the United States before the invention thereof by the applicant for patent, or on an international application by another who has fulfilled the requirements of paragraphs (1), (2), and (4) of section 371(c) of this title before the invention thereof by the applicant for patent.

The changes made to 35 U.S.C. 102(e) by the American Inventors Protection Act of 1999 (AIPA) and the Intellectual Property and High Technology Technical Amendments Act of 2002 do not apply when the reference is a U.S. patent resulting directly or indirectly from an international application filed before November 29, 2000. Therefore, the prior art date of the reference is determined under 35 U.S.C. 102(e) prior to the amendment by the AIPA (pre-AIPA 35 U.S.C. 102(e)).

2. Claims 1-4, 20, 27-28, 33, 36-37, 39, 41-47, 53-54, and 60-61 are rejected under 35 U.S.C. 102(e) as being anticipated by DiMambro (U.S. 7076545).

3. Regarding claim 1, DiMambro disclosed a method for processing packets through a plurality of protocol layers comprising:

accessing a packet associated with a connection (DiMambro, col. 1, lines 39-40); and

processing said packet through said plurality of protocol layers using a single thread from a single processor by assigning said connection to a single processor of a multiprocessor server system for processing wherein packets associated with said connection are directed to said single thread in said single processor for processing and wherein connection state information used by said plurality of protocol layers is preserved by mutual exclusion of other threads processing packets for said connection through said plurality of protocol layers (DiMambro, col. 1, lines 41-50. col. 3, lines 3-15, each flow may be assigned to a service queue that corresponds to a processor for protocol processing, col. 5, lines 5-7, management of each service queue is independently performed of the other queues).

4. Regarding claim 2, DiMambro disclosed the limitations as described in claim 1, including wherein said single thread is uninterrupted while processing said packet through said plurality of protocol layers (DiMambro, col. 2, lines 49-58).

5. Regarding claim 3, DiMambro disclosed the limitations as described in claim 1, including assigning said packet to a processing queue wherein said processing queue provides single threaded processing of said packet through said plurality of protocol layers (DiMambro, col. 1, lines 41-49).

6. Regarding claim 4, DiMambro disclosed the limitations as described in claim 3, including wherein said processing queue provides single threaded processing of said packet through said plurality of protocol layers by assigning only one packet to be processed by said plurality of protocol layers at a time (DiMambro, col. 2, lines 49-58).

7. Regarding claim 20, DiMambro disclosed a method for processing packets comprising:

accessing a packet associated with a connection (DiMambro, col. 1, lines 39-40); and

assigning said packet to a processing queue associated with a single processor of a multiprocessor server system wherein said processing queue provides uninterrupted single threaded processing of said data packet associated with the connection through a plurality of protocol layers using a single thread of the single processor, wherein state information of the packet within the connection is preserved so as to mutually exclude other threads from processing packets of said connection through said plurality of protocol layers (DiMambro, col. 1, lines 35-50. col. 3, lines 3-15,

each flow may be assigned to a service queue that has a thread and corresponds to a processor for protocol processing, col. 5, lines 5-7, management of each service queue is independently performed of the other queues).

8. Regarding claim 27, DiMambro disclosed the limitations as described in claim 20, including wherein said plurality of protocol layers includes a TCP protocol layer (DiMambro, col. 2, lines 49-55).

9. Regarding claim 28, DiMambro disclosed the limitations as described in claim 20, including wherein said plurality of protocol layers includes an IP protocol layer (DiMambro, col. 2, lines 49-55).

10. Regarding claim 33, DiMambro disclosed the limitations as described in claim 23, including wherein said queue is associated with said connection data structure (DiMambro, col. 5, lines 18-25).

11. Regarding claim 36, DiMambro disclosed a multiprocessor server system comprising:

a plurality of processors for processing packets through a plurality of protocol layers (DiMambro, col. 1, lines 35-40);

a plurality of threads running in the plurality of processors (DiMambro, col. 1, lines 35-45);

a plurality of queues, each queue associated with a respective processor of said plurality of processors (DiMambro, col. 1, lines 41-45); and

a memory resident connection data structure for assigning packets of a connection to a queue of said plurality of queues for processing said packets of said connection using a single thread associated with the single queue of a corresponding processor of said plurality of processors (DiMambro, col. 1, lines 32-45, col. 4, lines 8-20, col. 5, lines 15-25).

12. Regarding claim 37, DiMambro disclosed the limitations as described in claim 36, including wherein said connections are TCP connections (DiMambro, col. 4, lines 49-51).

13. Regarding claim 39, DiMambro disclosed the limitations as described in claim 36, including wherein a processor of said plurality of processors processes a packet of its queue without interruption through said plurality of protocol layers except for scheduling another packet on its queue (DiMambro, col. 5, lines 1-7).

14. Regarding claim 41, DiMambro disclosed the limitations as described in claim 37, including wherein said connection data structure is established for a new connection upon receiving a new connection request and wherein said connection data structure comprises an identifier of a queue associated with the single thread of the same

corresponding processor to which all packets of said new connection are to be assigned (DiMambro, col. 1, lines 32-45, col. 4, lines 8-20, col. 5, lines 15-25).

15. Regarding claim 42, DiMambro disclosed the limitations as described in claim 36, including a plurality of cache memories, each cache associated with a respective processor of said plurality of processors (DiMambro, col. 1, lines 39-45).

16. Regarding claim 43, DiMambro disclosed the limitations as described in claim 36, including wherein state information of any given packet of a same connection is preserved because said packets of said same connection are individually mutually excluded from said protocol layers (DiMambro, col. 1, lines 45-50).

17. Regarding claim 44, DiMambro disclosed a computer system comprising a processor coupled to a bus and a memory coupled to said bus and comprising instructions (DiMambro, col. 2, lines 24-35) that when executed implement a method for processing data packets comprising:

accessing a packet associated with a connection (DiMambro, col. 1, lines 39-40); and

processing said packet through said plurality of protocol layers using a single thread from a single processor by assigning said connection for processing to a single processor of a multiprocessor server system wherein packets associated with said connection are directed to said single thread associated with said single processor for

processing, wherein connection state information used by said plurality of protocol layers is preserved by mutual exclusion of other threads processing packets for said connection through said plurality of protocol layers (DiMambro, col. 1, lines 35-50. col. 3, lines 3-15, each flow may be assigned to a service queue that has a thread and corresponds to a processor for protocol processing, col. 5, lines 5-7, management of each service queue is independently performed of the other queues).

18. Regarding claim 45, DiMambro disclosed the limitations as described in claim 44, including wherein said single thread is uninterrupted while processing said packet through said plurality of protocol layers (DiMambro, col. 2, lines 43-47).

19. Regarding claim 46, DiMambro disclosed the limitations as described in claim 44, including wherein said packet are assigned to a processing queue wherein said processing queue provides single threaded processing of said packet through said plurality of protocol layers (DiMambro, col. 1, lines 38-45, col. 2, lines 43-47).

20. Regarding claim 47, DiMambro disclosed the limitations as described in claim 46, including wherein said processing queue provides single threaded processing of said packet through said plurality of protocol layers by assigning only one packet to be processed by said plurality of protocol layers at a time (DiMambro, col. 2, lines 49-58).

21. Regarding claim 53, DiMambro disclosed a computer system comprising a processor coupled to a bus and a memory coupled to said bus and comprising instructions (DiMambro, col. 2, lines 24-35) that when executed implement a method for processing data packets comprising:

accessing a packet associated with a connection (DiMambro, col. 1, lines 39-40); and

assigning said packet to a processing queue associated with a single processor of a multi processor server system, wherein said processing queue provides uninterrupted single threaded processing of said data packet associated with the connection through a plurality of protocol layers using a single thread of the single processor, wherein state information of the packet within the connection is preserved so as to mutually exclude other threads from processing packets of said connection through said plurality of protocol layers (DiMambro, col. 1, lines 35-50. col. 3, lines 3-15, each flow may be assigned to a service queue that has a thread and corresponds to a processor for protocol processing, col. 5, lines 5-7, management of each service queue is independently performed of the other queues).

22. Regarding claim 54, DiMambro disclosed the limitations as described in claim 53, including wherein said processing queue provides mutual exclusion of same-connection packet processing through said plurality of protocol layers (DiMambro, col. 5, lines 5-7).

23. Regarding claim 60, DiMambro disclosed the limitations as described in claim 53, including wherein said plurality of protocol layers includes a TCP protocol layer (DiMambro, col. 4, lines 49-51).

24. Regarding claim 61, DiMambro disclosed the limitations as described in claim 53, including wherein said plurality of protocol layers includes an IP protocol layer (DiMambro, col. 4, lines 49-51).

25. Claims 1-5, 20, 27-28, 44-48, 53-54, 60-61 are rejected under 35 U.S.C. 102(b) as being anticipated by Chang et al. (U.S. 6,338,078).

26. Regarding claim 1, Chang disclosed a method for processing packets through a plurality of protocol layers comprising:

accessing a packet associated with a connection (Chang, col. 2, lines 56-58); and

processing said packet through said plurality of protocol layers using a single thread from a single processor by assigning said connection to a single processor of a multiprocessor server system for processing wherein packets associated with said connection are directed to said single thread in said single processor for processing and wherein connection state information used by said plurality of protocol layers is preserved by mutual exclusion of other threads processing packets for said

connection through said plurality of protocol layers (Chang, Fig. 3, queues 62,64,66, 68 having corresponding processors CPU's 54,56,58,60; col. 2, lines 55-60, the packets are distributed to N high priority threads; col. 2, lines 64-67, Packets are distributed by a hashing function, hashing addresses of the packets to determine which queue/CPU pair it should be sent to, ; col. 4, lines 40-45, protocol processing; col. 4, line 60, single thread processing col. 5, lines 60-65; col. 6, lines 45-50, each CPU handles a particular queue).

27. Regarding claim 2, Chang disclosed the limitations as described in claim 1, including wherein said single thread is uninterrupted while processing said packet through said plurality of protocol layers (Chang, col. 5, lines 30-33, lines 60-66).

28. Regarding claim 3, Chang disclosed the limitations as described in claim 1, including assigning said packet to a processing queue wherein said processing queue provides single threaded processing of said packet through said plurality of protocol layers (Chang, col. 6, lines 44-47).

29. Regarding claim 4, Chang disclosed the limitations as described in claim 3, including wherein said processing queue provides single threaded processing of said packet through said plurality of protocol layers by assigning only one packet to be processed by said plurality of protocol layers at a time (Chang, col. 6, lines 45-50).

30. Regarding claim 5, Chang disclosed the limitations as described in claim 4, including wherein said packet is assigned to said processing queue based on address information of said connection (Chang, col. 2, lines 64-67).

31. Regarding claim 20, Chang disclosed a method for processing packets comprising:

accessing a packet associated with a connection (Chang, col. 2, lines 56-58); and

assigning said packet to a processing queue associated with a single processor of a multiprocessor server system wherein said processing queue provides uninterrupted single threaded processing of said data packet associated with the connection through a plurality of protocol layers using a single thread of the single processor, wherein state information of the packet within the connection is preserved so as to mutually exclude other threads from processing packets of said connection through said plurality of protocol layers (Chang, Fig. 3, queues 62,64,66, 68 having corresponding processors CPU's 54,56,58,60; col. 2, lines 55-60, the packets are distributed to N high priority threads; col. 2, lines 60-67, Packets are distributed by a hashing function, hashing addresses of the packets to determine which queue/CPU pair it should be sent to, ; col. 4, lines 40-45, protocol processing; col. 4, line 60, single thread processing col. 5, lines 60-65; col. 6, lines 45-50, each CPU handles a particular queue).

32. Regarding claim 27, Chang disclosed the limitations as described in claim 20, including wherein said plurality of protocol layers includes a TCP protocol layer (Chang, col. 4, lines 40-45).

33. Regarding claim 28, Chang disclosed the limitations as described in claim 20, including wherein said plurality of protocol layers includes an IP protocol layer (Chang, Fig. 4, 74).

34. Regarding claim 44, Chang disclosed a computer system comprising a processor coupled to a bus and a memory coupled to said bus and comprising instructions (Chang, col. 3, lines 25-55) that when executed implement a method for processing data packets comprising:

accessing a packet associated with a connection (Chang, col. 2, lines 56-58); and

processing said packet through said plurality of protocol layers using a single thread from a single processor by assigning said connection for processing to a single processor of a multiprocessor server system wherein packets associated with said connection are directed to said single thread associated with said single processor for processing, wherein connection state information used by said plurality of protocol layers is preserved by mutual exclusion of other threads processing packets for said connection through said plurality of protocol layers (Chang, Fig. 3, queues 62,64,66, 68 having corresponding processors CPU's 54,56,58,60; col. 2, lines 55-60, the packets

are distributed to N high priority threads; col. 2, lines 64-67, Packets are distributed by a hashing function, hashing addresses of the packets to determine which queue/CPU pair it should be sent to, ; col. 4, lines 40-45, protocol processing; col. 4, line 60, single thread processing col. 5, lines 60-65; col. 6, lines 45-50, each CPU handles a particular queue).

35. Regarding claim 45, Chang disclosed the limitations as described in claim 44, including wherein said single thread is uninterrupted while processing said packet through said plurality of protocol layers (Chang, col. 5, lines 30-33, lines 60-66)).

36. Regarding claim 46, Chang disclosed the limitations as described in claim 44, including wherein said packet are assigned to a processing queue wherein said processing queue provides single threaded processing of said packet through said plurality of protocol layers (Chang, col. 4, lines 60-67).

37. Regarding claim 47, Chang disclosed the limitations as described in claim 46, including wherein said processing queue provides single threaded processing of said packet through said plurality of protocol layers by assigning only one packet to be processed by said plurality of protocol layers at a time (Chang, col. 4, lines 60-67).

38. Regarding claim 48, hang disclosed the limitations as described in claim 47, including wherein said packet is assigned to said processing queue based on address information of said connection (Chang, col. 2, lines 64-67).

39. Regarding claim 53, Chang disclosed a computer system comprising a processor coupled to a bus and a memory coupled to said bus and comprising instructions (Chang, col.) that when executed implement a method for processing data packets comprising:

accessing a packet associated with a connection (Chang, col. 2, lines 56-58); and

assigning said packet to a processing queue associated with a single processor of a multi processor server system, wherein said processing queue provides uninterrupted single threaded processing of said data packet associated with the connection through a plurality of protocol layers using a single thread of the single processor, wherein state information of the packet within the connection is preserved so as to mutually exclude other threads from processing packets of said connection through said plurality of protocol layers (Chang, Fig. 3, queues 62,64,66, 68 having corresponding processors CPU's 54,56,58,60; col. 2, lines 55-60, the packets are distributed to N high priority threads; col. 2, lines 64-67, Packets are distributed by a hashing function, hashing addresses of the packets to determine which queue/CPU pair it should be sent to, ; col. 4, lines 40-45, protocol processing; col. 4, line 60, single

thread processing col. 5, lines 60-65; col. 6, lines 45-50, each CPU handles a particular queue).

40. Regarding claim 54, Chang disclosed the limitations as described in claim 53, including wherein said processing queue provides mutual exclusion of same-connection packet processing through said plurality of protocol layers (Chang, col. 4, lines 60-67).

41. Regarding claim 60, Chang disclosed the limitations as described in claim 53, including wherein said plurality of protocol layers includes a TCP protocol layer (Chang, col. 4, lines 40-45).

42. Regarding claim 61, Chang disclosed the limitations as described in claim 53, including wherein said plurality of protocol layers includes an IP protocol layer (Chang, col. 4, lines 40-45).

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

43. Claim 14 is rejected under 35 U.S.C. 103(a) as being unpatentable over Chang et al. (U.S. 6,338,078).

44. Regarding claim 14, Chang disclosed the limitations as described in claim 3. Chang did not explicitly state wherein the queue is an squeue.

However, it would have been obvious to one of ordinary skill in the art at the time the invention was made to substitute the queue of Chang with any type of queue, including an squeue, as long as the same functionality is performed, i.e. in order to achieve the predictable result of the queue queuing packets for processing. The type of queue is a matter of design choice since the same functionality is performed.

45. Claims 6-8, 19, 23-26, 33, 36-39, 41-43, 49-51, and 56-59 are rejected under 35 U.S.C. 103(a) as being unpatentable over Chang et al. (U.S. 6,338,078) in view of Syvanne (U.S. 2002/0112188).

46. Regarding claims 6, 23, 49, and 56, Chang disclosed the limitations as described in claims 1, 20, 44, and 53. Chang also disclosed that packets are distributed to one of the N queues by using a hashing function (Chang, col. 2, lines 64-66).

Chang did not explicitly state generating a unique connection data structure specific to said connection stored in said packet associated with the connection.

In an analogous art, Syvanne disclosed a method for handling information about packet data connections in which connection data structures are created (Syvanne,

[0009]). As such, Syvanne suggests the use of data structures for classifying packets and shows that such packet classification was well known at the time the invention was made.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate using data structures to classify packets according to connections into the system of Chang in order to obtain the predictable result of distributing packets to its corresponding queue/processor duo based on the connection that the packet belongs, thereby providing a more scalable system allowing multiple ways of classifying packets.

47. Regarding claims 7, 24, 50, and 57, Chang and Syvanne disclosed the limitations as described in claims 6, 23, 49, and 56, including wherein said address information comprises a local IP address and a remote IP address (Chang, Fig. 4, 74). See motivation above.

48. Regarding claims 8, 25, 51, 58, Chang and Syvanne disclosed the limitations as described in claims 7, 24, 49, 57, including wherein said address information further comprises a remote port address and a local port address (Chang, Fig. 4, 76). See motivation above.

49. Regarding claims 19, 26, 59, Chang and Syvanne disclosed the limitations as described in claim 6, 25, and 58. Chang and Syvanne did not explicitly state wherein

subsequent data packets of said connection are assigned to said single processor based on said connection data structure.

However as shown above, Syvanne suggests the use of data structures for classifying packets and shows that such packet classification was well known at the time the invention was made.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate using data structures to classify packets according to connections into the system of Chang in order to obtain the predictable result of distributing packets to its corresponding queue/processor duo based on the connection that the packet belongs, thereby providing a more scalable system allowing multiple ways of classifying packets.

50. Regarding claim 33, Chang and Syvanne disclosed the limitations as described in claim 23. Chang and Syvanne did not explicitly state wherein said queue is associated with said connection data structure.

However, it would have been obvious to one of ordinary skill in the art at the time the invention was made that using the connection data structure to properly classify the packets and properly distribute them to the queue/processor duo includes an association between the connection data structure and the queue. See motivation above.

51. Regarding claim 36, Chang disclosed a multiprocessor server system comprising:

a plurality of processors for processing packets through a plurality of protocol layers (Chang, col. 2, lines 55-56);

a plurality of threads running in the plurality of processors (Chang, col. 2, lines 55-60, "N high priority threads"); and

a plurality of queues, each queue associated with a respective processor of said plurality of processors (Chang, col. 2, lines 59-65).

Chang also disclosed assigning packets of a connection to a queue of said plurality of queues for processing said packets of said connection using a single thread associated with the single queue of a corresponding processor of said plurality of processors (Chang, Fig. 3, queues 62,64,66, 68 having corresponding processors CPU's 54,56,58,60; col. 2, lines 55-60, the packets are distributed to N high priority threads; col. 2, lines 64-67, Packets are distributed by a hashing function, hashing addresses of the packets to determine which queue/CPU pair it should be sent to, ; col. 4, lines 40-45, protocol processing; col. 4, line 60, single thread processing col. 5, lines 60-65; col. 6, lines 45-50, each CPU handles a particular queue).

Chang did not explicitly state including a memory resident connection data structure for such assignment.

In an analogous art, Syvanne disclosed a method for handling information about packet data connections in which connection data structures are created (Syvanne, [0009]). As such, Syvanne suggests the use of data structures for classifying packets

and shows that such packet classification was well known at the time the invention was made.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate using data structures to classify packets according to connections into the system of Chang in order to obtain the predictable result of distributing packets to its corresponding queue/processor duo based on the connection that the packet belongs, thereby providing a more scalable system allowing multiple ways of classifying packets.

52. Regarding claim 37, Chang and Syvanne disclosed the limitations as described in claim 36, including wherein the connections are TCP connections (Chang, col. 2, lines 65-67). See motivation above.

53. Regarding claim 38, Chang and Syvanne disclosed the limitations as described in claim 37, including wherein said plurality of protocol layers comprise IP, TCP, and socket layers (Chang, Fig. 4). See motivation above.

54. Regarding claim 39, Chang and Syvanne disclosed the limitations as described in claim 36, including wherein a processor of said plurality of processors processes a packet of its queue without interruption through said plurality of protocol layers except for scheduling another packet on its queue (Chang, col. 2, lines 55-67). See motivation above.

55. Regarding claim 41, Chang and Syvanne disclosed the limitations as described in claim 37, including wherein said connection data structure is established for a new connection upon receiving a new connection request and wherein said connection data structure comprises an identifier of a queue associated with the single thread of the same corresponding processor to which all packets of said new connection are to be assigned (See rejection for claim 19, see also Chang, Fig. 3, queues 62,64,66, 68 having corresponding processors CPU's 54,56,58,60; col. 2, lines 55-60, the packets are distributed to N high priority threads; col. 2, lines 60-67, Packets are distributed by a hashing function, hashing addresses of the packets to determine which queue/CPU pair it should be sent to, ; col. 4, lines 40-45, protocol processing; col. 4, line 60, single thread processing col. 5, lines 60-65; col. 6, lines 45-50, each CPU handles a particular queue). See motivation above.

56. Regarding claim 42, Chang and Syvanne disclosed the limitations as described in claim 36, including a plurality of cache memories, each cache associated with a respective processor of said plurality of processors (Chang, Fig. 3, 62, 64, 66, 68). See motivation above.

57. Regarding claim 43, Chang and Syvanne disclosed the limitations as described in claim 36, including wherein state information of any given packet of a same connection is preserved because said packets of said same connection are individually

mutually excluded from said protocol layers (Chang, col. 2, lines 55-67). See motivation above.

(10) Response to Argument

Before a detailed response to Applicant's arguments, Examiner would like to remind Applicant to correct the "Cross Reference to Related Applications" section of Applicant's Specification for being incomplete as indicated in the previous office action. Correction is required. See MPEP § 608.01(b).

For each 35 U.S.C. 102 rejection, Applicant presents two principal arguments:

a) Applicant argues, "None of the references cited by Examiner teach assigning 'a connection to a single processor,' and claim 1 further requires that 'wherein packets associated with said connection are directed to said single thread in said single processor for processing' " [Br p10].

b) Applicant argues that the prior art did not disclose "wherein connection state information used by said plurality of protocol layers is preserved by mutual exclusion of other threads from processing packets for said connection" [Br 10].

For the 35 U.S.C. 103 rejections, Applicant presents a single argument:

c) Applicant argues that Chang fails to teach "generating a unique connection data structure specific to said connection" [Br p 19]

Response to Argument a) regarding DiMambro

In addition to the argument a) above, Applicant argues that the DiMambro reference teaches away from the method of claim 1 because Applicant believes that DiMambro does not assign all processing of received packets for a connection to a single thread [Br 11].

Examiner respectfully disagrees for the following reasons:

DiMambro explicitly states, "Each queue has an associated service thread or process that performs upper layer protocol processing of the packet (e.g., for IP and TCP)" (col. 1, lines 41-43).

DiMambro also explicitly states, "Thus, all packets in one flow or connection may traverse the same service queue" (col. 1, lines 46-48).

This clearly shows that packets of a single connection are all assigned to the same service thread.

For further evidence, DiMambro provides more detail for how the system identifies a particular service queue for a particular packet by disclosing, "The, the interface may configure and pass with the packet a flow identifier for identifying a communication flow, a processor identifier for identifying a particular processor of the computer system, or some other identifier or characteristic of the packet. The identifier may be used to directly identify a service queue, or may be hashed or otherwise manipulated to obtain a value usable in selecting a queue (col. 3, lines 12-21).

DiMambro provides a specific embodiment where each queue is only associated with a single processor. DiMambro recites, "In one embodiment of the invention, one

service thread or process is instantiated for each service queue, and only services a single assigned queue" (col. 3, lines 25-28).

It is clearly shown by these mappings that DiMambro disclosed at least one embodiment for load balancing of connections to separate threads (i.e. all packets from a particular connection are directed to the same thread).

Therefore it is absolutely clear that DiMambro processes a packet through a plurality of protocol layers using a single thread from a single processor by assigning the connection that the packet is part of to a single processor of a multiprocessor server system, as claimed.

Response to Argument b) regarding DiMambro

Regarding Argument b):

Firstly, Applicant does not present any substantive arguments to distinguish the limitations of the claim from the cited prior art relied upon in the rejection. Instead, Applicant merely: (1) reproduce the portions of the cited reference relied upon by the Examiner, and (2), assert that the language of the claim is not taught.

Therefore, the Applicant has failed to rebut the Examiner's rejection of the claim with any persuasive analysis. Instead, Applicant grounds their argument on a conclusory assertion. This form of argument is wholly ineffective in demonstrating error in the Examiner's prima facie case to establish the patentability of the claims. *See Ex parte Belinne*, No. 2009-004693, slip op. at 7-8 (BPAI Aug 10, 2009)(informative), *available at* http://www.uspto.gov/web/offices/dcom/bpai/its/fd09004693.pdf

Examiner further notes that a general allegation that the claims define a patentable invention without specifically pointing out how the language of the claims patentably distinguishes them from the reference does not comply with the requirements of 37 C.F.R. 1.111(b).

Secondly, Examiner points out that the limitation, "wherein connection state information used by said plurality of protocol layers is preserved by mutual exclusion of other threads from processing packets for said connection through said plurality of protocol layers" appears to simply be a result of what occurs when the packets of a connection are all directed to the same processor for protocol processing. Accordingly, the limitation should not be given patentable weight.

However, for argument sake, assuming the limitation is deserved patentable weight, the Examiner contends that the reference does in fact teach this limitation for the following reasons.

Examiner points out that "connection state information" is simply defined as the context needed to manage the protocol processing of a connection. As such, connection state information is required in order to properly protocol process packets through layers.

Applicant's Background section recites that prior art systems that use multiple processors to process packets of a single connection suffer from drawbacks, one these drawbacks being that "packets for the same connection go through various protocol layers where they have to contend for access to their state structures at each layer"

(Applicant's Specification, pages 2-3). Applicant's solution for this is a system where all packets from a particular connection are processed by the same thread/processor. Independent claim 1 calls for assigning the packets of a particular connection to a single processor for protocol processing.

As Applicant's specification shows, it is inherent that connection state information must be used in order to properly perform protocol processing over multiple layers.

As clearly shown above, DiMambro disclosed this functionality in that every packet for a particular connection is directed to the same queue/processor pair (or thread) for protocol processing of multiple layers such as the upper layer protocol processing of the packet (e.g., for IP and TCP)" (col. 1, lines 41-43) as mentioned above. Therefore the single thread performs the protocol processing for multiple layers and as such the required connection state information is mutually excluded from other threads since it is only the single thread performing the protocol processing for this connection. Since each thread manages its own connection, all of the connection state information is isolated from the other threads. Examiner also states that it would be pointless for a thread that is not protocol processing the packets of a connection to be able to access that connection's state information as there is absolutely no use of this connection state information by the thread.

Therefore the rejection is respectfully maintained.

Applicant relies on the same arguments for claims 2-5, 20, 27-28, 33, 36-37, 39, 41-47, 53-54, 60, and 61, and therefore the rejections for these claims are also respectfully maintained.

Response to Argument a) regarding Chang

Firstly, Examiner points out that throughout Applicant's arguments, Applicant mentions the use of a data structure called a 'conn'. This data structure is not disclosed in the independent claims and therefore Applicant's arguments with respect to this data structure are moot.

Regarding Argument a) that Chang fails to teach assigning packets of a single connection to a single thread, Examiner respectfully disagrees.

Chang explicitly states, "Packets are distributed to one of the N queues by using a hashing function based on the source MAC address, source IP address, or the packet's source and destination TCP port number, or all or a combination of the foregoing" (Chang, col. 2, lines 64-67). The hashing function is represented as element 50 in Figure 3. Chang further explains that each processor (Fig. 3, 54, 56, 58, 60) has its own corresponding queue to hold the incoming packets (Fig. 3, 62, 64, 66, 68). Chang further disclosed that the entire purpose for this hashing function is "so that packets associated with a given device (i.e. the source device) will be handled in the same particular queue. Therefore no matter which CPU 54-60 handles a particular queue, the packets associated with a particular device will flow to one of the applications 52-56 in sequence. Therefore it can be clearly seen that the entire purpose of Chang is to assign a connection to a particular queue/CPU pair so that this particular processor handles all incoming packets for this connection. The hashing function hashes the connection parameters for every packet, and it is clearly understood that

every packet from the same connection will be sent to the same queue/CPU pair since the hash value for every single packet from this connection will be the same. Therefore every packet for the same connection is assigned to the same queue/CPU pair.

As Applicant points out "the reference must show or suggest that all packets associated with a particular connection are sent to a single thread in a single processor" [Br 10]. Chang explicitly disclosed that each queue/CPU pair has one thread. At col. 5, lines 20-25, Chang disclosed, "in accordance with the invention depicted in Fig. 3, this IP queue is concurrently processed by multiple threads with one thread per CPU, and one queue per thread. Therefore it is clearly shown by Chang that all packets of the same connection are assigned to the same thread of the same queue/CPU pair as depicted in Figure 3 of the invention.

Therefore it is absolutely clear that Chang processes a packet through a plurality of protocol layers using a single thread from a single processor by assigning the connection that the packet is part of to a single processor of a multiprocessor server system, as claimed.

Response to Argument b) regarding Chang

Regarding Argument b), the Examiner respectfully disagrees.

The Examiner points out that "connection state information" is simply defined as the context needed to manage the protocol processing of a connection. As such, connection state information is required in order to properly protocol process packets through layers.

Applicant's Background section recites that prior art systems that use multiple processors to process packets of a single connection suffer from drawbacks, one these drawbacks being that "packets for the same connection go through various protocol layers where they have to contend for access to their state structures at each layer" (Applicant's Specification, pages 2-3). Applicant's solution for this is a system where all packets from a particular connection are processed by the same thread/processor. Independent claim 1 calls for assigning the packets of a particular connection to a single processor for protocol processing.

As Applicant's specification shows, it is inherent that connection state information must be used in order to properly perform protocol processing over multiple layers.

As clearly shown in the rejection, Chang clearly disclosed that packets for a connection will always be processed by the single thread (i.e. queue/CPU pair). Chang also clearly disclosed the single thread performing protocol processing with specific example towards TCP/IP (col. 4, lines 40-50). Therefore the single thread performs the protocol processing for multiple layers and as such the required connection state information is mutually excluded from other threads since it is only the single thread performing the protocol processing for this connection. Since each thread manages its own connection, all of the connection state information is isolated from the other threads. Examiner also states that it would be pointless for a thread that is not protocol processing the packets of a connection to be able to access that connection's state information as there is absolutely no use of this connection state information by the thread.

Therefore the rejection is respectfully maintained.

Applicant relies on the same arguments for claims 2-8, 14, 19-20, 23-28, 33, 36-39, 41-47, 49-51, 53-54, 56-60, and 61, and therefore the rejections for these claims are also respectfully maintained.

Response to Argument c)

In response to applicant's arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.,* 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986).

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

/J Bret Dennison/

Primary Examiner, Art Unit 2443

Conferees:

/Kenny S Lin/

Primary Examiner, Art Unit 2452

/George C Neurauter, Jr./

Primary Examiner, Art Unit 2443